REMARKS

Applicant is in receipt of the Office Action mailed January 6, 2006. Claims 1-25, 32-44, 46-60, and 62-89 were rejected. There are no current amendments to the claims. Reconsideration of the case is earnestly requested in light of the following remarks.

Section 112 Rejections

Claims 16, 71, 80-81, and 87 were rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the written description requirement. Applicant respectfully traverses this rejection.

Independent claim 87 recites as follows:

> 87. (Previously Presented) A computer-implemented method for creating a graphical program, the method comprising:
>
> creating a first portion of graphical code in response to user input, wherein the first portion of graphical code comprises one or more nodes that visually indicate functionality for responding to programmatic events generated during execution of the graphical program; and
>
> configuring the graphical program to dynamically register a first programmatic event during execution of the graphical program, wherein dynamically registering the first programmatic event comprises dynamically associating the first programmatic event with the first portion of graphical code;
>
> wherein, before said dynamically registering the first programmatic event during execution of the graphical program, the first portion of graphical code does not execute in response to the first programmatic event being generated;
>
> wherein said dynamically registering the first programmatic event causes the first portion of graphical code to execute in response to the first programmatic event being generated.

The Examiner rejected claim 87 because of the negative limitation of, "wherein, before said dynamically registering the first programmatic event during execution of the graphical program, the first portion of graphical code does not execute in response to the first programmatic event being generated". However, Applicant respectfully submits that it is clear from the specification that the first portion of graphical code does not execute in response to the first programmatic event being generated until after the first programmatic event has been dynamically registered. For example, p. 9, line 28 – p. 10, line 18 of the Summary of the Invention describes that:

When using a dialog such as described above, the specified events may be registered such that upon execution startup, the graphical program begins receiving and responding to the specified events. However, in some cases it may be desirable to dynamically register an event during execution of the graphical program. For example, for a given event, it may only be necessary or desirable to receive and respond to the event if some condition becomes true during the course of executing the program. Thus, the event may be dynamically registered when the condition becomes true or at some pre-determined point in the program. In one embodiment, an event registration node may be included in the block diagram and may be configured to register an event. The event registration node may be executable to dynamically register the event such that, after registering the event, the graphical program is operable to receive and respond to the event.

Similarly, it may also be desirable to un-register an event at some point during execution of the graphical program. The event to be un-registered may be an event that was specified at edit time via a dialog as described above or may be an event that was previously registered dynamically. In one embodiment, an event un-registration node may be included in the block diagram and may be configured to un-register an event. The event un-registration node may be executable to dynamically un-register the event such that, after un-registering the event, the graphical program does not receive and respond to the event.

Thus, it is clear to those skilled in the art that the first portion of graphical code does not execute in response to the first programmatic event being generated until after the first programmatic event has been dynamically registered at some point during execution of the graphical program. The purpose of dynamically registering the first programmatic event is to enable the first portion of graphical code to execute in response to the first programmatic event being generated.

Applicant thus respectfully submits that the Section 112 rejection of claim 87 should be removed. Claims 16 and 71 were rejected for similar reasons, and thus, Applicant respectfully submits that these rejections should also be removed.

Claim 80 recites as follows:

80. (Previously Presented) The method of claim 1,

wherein said receiving the third user input explicitly specifying the one or more user interface events to configure for the first node does not include receiving user input specifying a connection between the first node and a second node.

The Examiner rejected claim 80 because of the negative limitation. However, the specification clearly describes an embodiment in which receiving the third user input explicitly specifying the one or more user interface events to configure for the first node does not include receiving user input specifying a connection between the first node and a second node. For example, Figure 11 illustrates an exemplary event configuration dialog. As described on p. 28, line 26 – p. 30, line 12, in one embodiment the user may interact with such an event configuration dialog to configure a set of events for an event structure node. Thus, it is clear to those skilled in the art that in one embodiment, the user interface events to configure for the first node may be specified by user input to a dialog box or other graphical user interface, without requiring the user to specify a connection between the first node and a second node.

Applicant thus respectfully submits that the Section 112 rejection of claim 80 should be removed. Claim 81 was rejected for similar reasons, and thus, Applicant respectfully submits that this rejection should also be removed.

Section 102 Rejections

Claims 1-7, 10-25, 32-44, 46-60, 62-79, and 82-89 were rejected under 35 U.S.C. 102(b) as being anticipated by Cain et al., U.S. Patent No. 5,651,108 (hereinafter "Cain"). Applicant respectfully traverses this rejection.

Cain relates generally to a relational database management system (RDBMS) in which the user places screen objects (e.g, boxes, screen buttons, table objects, and the like) on an on-screen window or "form". The screen objects have associated properties, as well as methods that execute in response to events.

As per claim 1, Cain does not teach several of the limitations recited therein. In particular, Cain does not teach, "receiving third user input explicitly specifying one or more user interface events to configure for the first node".

In the rejection of claim 1, the Examiner refers to the section at Col. 10, line 48 – Col. 13, line 17, in which Cain describes an example of building a program which provides a customized button that responds to a mouse click event by displaying a dialog box. The button includes built-in methods that execute in response to events (Col. 3, lines 45-55 and Col. 11, lines 55-59). In particular, the button includes a built-in

"pushButton" method that executes in response to push-button events which are generated when the user clicks the button during execution of the program. Cain teaches that the user customizes the pushButton method to make it respond to a push-button event by displaying a dialog box (Col. 12, lines 16-18).

As per the limitation of, "receiving third user input explicitly specifying one or more user interface events to configure for the first node," the Examiner refers to the description of FIGS. 4D-4E. The Examiner asserts that, "users can change or attach new user interface events which the button will respond to via input on the graphical user interface". Applicant respectfully disagrees.

FIG. 4D illustrates a popup menu from which the programmer can select a button property, such as Button Type, Center Label, or Design, or select the Methods menu item (Col. 11, lines 30-44). Selecting the Methods menu item causes the Methods window shown in FIG. 4E to be displayed. The Methods window includes a Built-in Methods listbox which lists the built-in (default) methods already attached to the button (Col. 11, lines 62-67) and a Custom Methods box which lists the names of additional custom (user-defined) methods that have been attached to the button (Col. 12, lines 7-9). Cain teaches that, "The behavior of the button object can be varied in two ways: the built-in methods can be edited or new custom methods can be written" (Col. 12, lines 14-16). Thus, the graphical user interface in FIG. 4E allows the user to change the built-in methods attached to the button or attach new custom methods to the button, but Cain teaches nothing about changing or attaching new user interface events. Applicant respectfully submits that the Examiner has confused a method with an event.

Claim 1 recites, "receiving third user input explicitly specifying one or more user interface events to configure for the first node" and "configuring the first node to receive the one or more user interface events explicitly specified by the third user input during execution of the graphical program". Cain clearly teaches that the button is configured to receive certain events, such as the push-button event, by default. Cain does not teach configuring the button to receive one or more events that have been explicitly specified by the user, as recited in claim 1.

Furthermore, Cain does not teach, "associating one or more portions of graphical code with the first node in response to fourth user input, wherein each portion of

graphical code comprises one or more nodes for responding to one or more of the user interface events which the first node is configured to receive." On the contrary, Cain teaches that the methods which respond to the events comprise text-based code, i.e., program code constructed in a text-based programming language, as opposed to graphical code which comprises one or more nodes. For example, as shown in FIGS. 4F – 4H, the pushButton method which executes to respond to the push-button event is constructed from text-based code, not graphical code.

For at least the reasons given above, Applicant respectfully submits that Cain does not teach numerous limitations recited in claim 1, and thus, claim 1 is patentably distinct over Cain. Since independent claims 19, 23, 32, 36, 53, 66, and 86 recite similar limitations as those discussed above with respect to claim 1, Applicant submits that these independent claims are also patentably distinct over Cain, for reasons similar to those given above.

As per independent claim 87, the method comprises dynamically registering the first programmatic event during execution of the graphical program. Before the first programmatic event is dynamically registered during execution of the graphical program, the first portion of graphical code does not execute in response to the first programmatic event being generated. Dynamically registering the first programmatic event enables the first portion of graphical code to execute in response to the first programmatic event being generated.

Cain does not teach dynamically registering the first programmatic event during execution of the graphical program. In the rejection of claim 87, the Examiner refers to the user interacting with the graphical user interface of FIG. 4E. As discussed above, the user interacts with this graphical user interface to change the built-in methods attached to the button or attach new custom methods to the button. The user interacts with the graphical user interface while editing the graphical program, not during execution of the graphical program. Cain teaches nothing at all about an event being dynamically registered during execution of the program.

Furthermore, as discussed above, Cain teaches executing text-based code in response to events being generated, such as the pushButton method shown in FIGS. 4F – 4H. Cain does not teach executing a portion of graphical code in response to an event

being generated, where the graphical code comprises one or more nodes that visually indicate functionality for responding to the event. For at least these reasons, Applicant respectfully submits that claim 87 is patentably distinct over Cain.

Since the independent claims have been shown to be patentably distinct, Applicant submits that the dependent claims are also patentably distinct over the cited art, for at least this reason. Applicant also submits that numerous ones of the dependent claims recite further distinctions over the cited art.

For example, as per claim 2, Cain does not teach, "wherein the first node comprises one or more sub-diagrams, wherein said associating the one or more portions of graphical code with the first node comprises displaying each portion of graphical code within one of the sub-diagrams of the first node." The Examiner has equated the first node with the button described above. However, the button does not comprise one or more sub-diagrams. Also, as discussed above, Cain does not teach associating one or more portions of graphical code with the button, wherein each portion of graphical code comprises one or more nodes for responding to one or more of the events which the button is configured to receive. Thus, Cain does not teach displaying such portions of graphical code within sub-diagrams of the button.

Applicant further submits that numerous ones of the other dependent claims recite further distinctions over the cited art. However, since the independent claims have been shown to be patentably distinct, a further discussion of the dependent claims is not necessary at this time.

Section 103 Rejections

Claims 8 and 9 were rejected under 35 U.S.C. 103(a) as being unpatentable over Cain and U.S. Patent No. 6,578,174 to Zizzo (hereinafter Zizzo). Applicant respectfully traverses these rejections.

Applicant reminds the Examiner that if an independent claim is non-obvious under 35 U.S.C. 103, then any claim depending therefrom is non-obvious. *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988). Applicant thus respectfully submits that since the independent claims have been shown above to be patentably distinct and non-

obvious over the prior art, dependent claims 8 and 9 are also patentably distinct and non-obvious, for at least this reason.

Applicant also submits that claims 8 and 9 recite further distinctions not taught or suggested by the cited references, taken either singly or in combination. However, since the independent claims have been shown to be patentably distinct, a further discussion of the dependent claims is not necessary at this time.

CONCLUSION

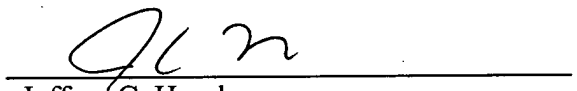
In light of the foregoing amendments and remarks, Applicant submits the application is now in condition for allowance, and an early notice to that effect is requested.

If any extensions of time (under 37 C.F.R. § 1.136) are necessary to prevent the above referenced application(s) from becoming abandoned, Applicant(s) hereby petition for such extensions. If any fees are due, the Commissioner is authorized to charge said fees to Meyertons, Hood, Kivlin, Kowert & Goetzel PC Deposit Account No. 50-1505/5150-58700/JCH.

Also enclosed herewith are the following items:

☒ Return Receipt Postcard

Respectfully submitted,



Jeffrey C. Hood
Reg. No. 35,198
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert & Goetzel PC
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8800
Date: 2/21/2006 JCH/JLB